## REMARKS

In view of the following remarks, Applicant respectfully requests reconsideration and allowance of the subject application. This amendment is fully responsive to all issues raised in the Office Action mailed October 11, 2005.

*Rejections Under 35 U.S.C. §112*

Applicant has amended certain claims to address the rejections based on a lack of antecedent basis.

Applicant traverses the rejection of claim 12 based on what appears to be an assertion of unclaimed essential matter. The Action fails to identify the subject matter the Examiner considers to be essential to the claim. Hence, Applicant traverses the rejection as lacking factual basis and requests clarification of the Examiner's position.

*Rejections Under 35 U.S.C. §102*

Claims 1-16 were rejected under 35 U.S.C. §102(b) as being anticipated by U.S. Patent No. 5,940,850 to Harish, et al (hereinafter, "Harish"). Applicants traverse these rejections.

*Claims 1-6*

Applicant submits that the Action fails to establish a *prima facie* case that Harish anticipates independent claim 1. Anticipation under 35 U.S.C. §102 requires that *each and every element* of the claim be set forth in the manner recited in the claim in a single prior art reference. (See, MPEP 2131). Independent claim 1 positively recites structural elements including an agent, a host, and a controller coupled to the agent. The Action sets forth no evidence or argument that Harish discloses (or even suggests) structural elements including an agent, a host, and a controller coupled to the agent, as recited in claim 1. Therefore, the Action fails to establish a *prima facie* case of anticipation.

Applicant further submits that Harish cannot anticipate (or render obvious) independent claim 1 because Harish neither discloses (nor even suggests) limitations recited in independent claim 1. Claim 1 recites "an agent coupled to a host, the agent having volatile memory for storing a first table, the table having entries to map the virtual storage segments to the storage locations; and a controller coupled to the agent, the controller having non-volatile memory for storing a second table, the controller intermittently causing contents of the first table to be replaced by contents of the second table." The Action asserts that Harish discloses this limitation, and cites the

Abstract and column 2, lines 10-19 and 25-34 to support the rejection.

Applicant disagrees. The cited text reads as follows:

> A system and method for loading dynamic data stored in read-only memory (ROM) is loaded into random access memory (RAM) only when it is being modified. Unmodified dynamic data is used from ROM saving valuable RAM space. Virtual memory page table entries are created for all dynamic data with the physical reference pointing to the dynamic data in ROM. Page table entries in a translation table for dynamic data in ROM include a virtual address to physical address mapping and are marked read-only causing a write-access exception if an attempt is made to write to or update the dynamic data. Write-access exceptions are intercepted, and a write-access exception caused by an attempt to write to dynamic data in ROM causes the system to allocate a dynamic data page in RAM, copy the ROM data to the RAM, update the page table entry to point to the RAM page rather than the ROM page, and finally to update the dynamic data now present in read-write RAM.
>
> ROM data is loaded into random access memory (RAM) only when that data is actually being modified. An attempt is detected to write to dynamic data in ROM and generates a write-access fault. The write-access fault is captured causing the system to copy the ROM data to RAM, change a page table entry to point to the RAM copy, and then write the data. This mechanism avoids loading ROM data that is not modified thereby reducing the RAM requirements for a given system.
>
> The present invention is directed to a computer implemented method for loading read-only memory data into random access memory only when the data is modified, in a computer system having a processor, random access memory and read-only memory, comprising the steps of: storing a data page table entry for translating a virtual data address into physical data address for each dynamic data page; storing in the page table an indicator that-the page table entry is read-only if the physical data address is in read-only memory; receiving a write-access exception whenever the computer system attempts to write to a dynamic data page having a read-only indicator, and loading the read-only memory dynamic data page into random access memory in response to the write-access exception. The page table entry for the data page is then updated for future access by the system.

Nothing in this text discloses (or even suggests) an agent coupled to a host, the agent having volatile memory for storing a first table, the table having entries to map the virtual storage segments to the storage locations; and a controller coupled to the agent, the controller having non-volatile memory for storing a second table, the controller intermittently causing contents of the first table to be replaced by contents of the second table, as explicitly recited in claim 1. Therefore, Harish cannot anticipate independent claim 1.

Applicant notes that claims 2-6 depend from independent claim 1, and are allowable at least by virtue of their dependency.

*Claims 7-11*

Applicant submits that the Action fails to establish a *prima facie* case that Harish anticipates independent claim 7. The Action asserts that claim 7 is essentially the system claim for claim 1. Applicant disagrees. Claim 7 sets forth limitations distinct from those recited in claim 1. The Action sets forth no evidence or argument that Harish discloses (or even suggests) limitations recited in claim 7. Therefore, the Action fails to establish a *prima facie* case of anticipation.

Claim 7 recites "a plurality of variables indicating states of an entry in the first table or the second table; and an offset for the entry, wherein the offset includes a logic unit number identifier and a block identifier." The Action asserts that Harish discloses these limitations, and cites column 2, lines 10-19, column 3, lines 56-67, and column 4, lines 20-37 to support the rejection. Applicant disagrees. The cited text reads as follows:

ROM data is loaded into random access memory (RAM) only when that data is actually being modified. An attempt is detected to write to dynamic data in ROM and generates a write-access fault. The write-access fault is captured causing the system to copy the ROM data to RAM, change a page table entry to point to the RAM copy, and then write the data. This mechanism avoids loading ROM data that is not modified thereby reducing the RAM requirements for a given system.

Virtual memory management systems separate the logical system memory reference from the physical address of the referenced object. This allows the system to reference a greater amount of memory than is physically present in a system. Virtual memory management schemes typically are based on pages of memory. At any point in time a number of pages are present in memory. These pages are tracked using a page table that cross references the virtual address to the location of the actual page containing the memory data. If an address reference is requested that does not exist in the page table, a page fault occurs requiring the referenced page to be loaded into memory. Memory pages are typically managed on a least recently used basis. The system will discard the least recently used page in memory and replace it with the one requested. If necessary, the page to be discarded is written out to storage before the page is freed.

The present invention adds logic to the routine for handling write-access exceptions or faults. The flowchart of FIG. 3 illustrates the process flow for handling write-access faults for dynamic data resident in ROM. The process begins at the start block 301. The system then raises a write-access fault at step 306 when an attempt is made to modify a piece of dynamic data at step 302. After an attempt to write dynamic data is made at step 302, a query is made of whether a page table entry (PTE) exists at block 303. If yes, the process continues to step 304 to be hereinafter described. If no PTE entry exists, flow exits to the right of block 303, indicating a process page fault at block 305 and the process returns to block 303. The system tests in step 304 to determine whether the page table entry PTE protection is "read-only". If not, the data is written at step 316 using known processes. If the entry is "read-only" a "write-access fault" is raised at step 306.

Nothing in this text discloses (or even suggests) a plurality of variables indicating states of an entry in the first table or the second table; and an offset for the entry, wherein the offset includes a logic unit number identifier and a block identifier, as recited in claim 7. Therefore, Harish cannot anticipate independent claim 7.

Claims 8-11 depend from independent claim 1, and are allowable at least by virtue of their dependency.

*Claims 12-16*

Applicant submits that the Action fails to establish a *prima facie* case that Harish anticipates independent claim 12. The Action asserts that claim 12 is anticipated by Harish, but lacks any analysis of claim 12 whatsoever. Therefore, the Action fails to establish a *prima facie* case of anticipation.

Claims 13-16 depend from independent claim 12, and are allowable at least by virtue of their dependency.

*Rejections Under 35 U.S.C. §103*

Claims 12-16 were rejected under 35 U.S.C. §103(a) as being unpatentable over Harish in view of U.S. Patent No. 5,483,649 to Kuznetzov (hereinafter, "Kuznetzov"). Applicant traverses this rejection, and asserts that the final Action fails to establish a *prima facie* case of obviousness. To establish a prima facie case of obviousness, the Action must establish that each element of the claim is disclosed or suggested by the cited references. *See*, MPEP 2142.

Independent claim 12 includes a limitation reciting "identifying portions of the virtual storage segment to be effected during a write operation." The Action asserts that Harish discloses this limitation, and cites column 2 lines 19-51 and column 3, lines 56-67 to support the assertion. Applicants disagree. Contrary to the assertion in the action, nothing in the cited text, excerpted above, discloses or suggests *identifying portions of the virtual storage segment to be effected during a write operation*, as recited in claim 12.

Claim 12 further recites limitations requiring "storing a record of the identified portions at a second table and not at the first table" and "writing to a second storage location, whereby the writing operation occurs at portions of the second storage location associated with the identified portions." The Action asserts that Harish discloses this limitation, and cites column 2 lines 19-51 and column 3, lines 56-67 to support the assertion. Applicants disagree. Contrary to the assertion in the action, nothing in the cited text, excerpted above, discloses or suggests storing a record of the identified portions at a second table and not at the first table and writing to a second

storage location, whereby the writing operation occurs at portions of the second storage location associated with the identified portions," as recited in claim 12.

Claims 13-16 depend from independent claim 12, and are allowable at least by virtue of their dependency.

## CONCLUSION

This application is in condition for allowance. Applicant respectfully requests reconsideration and prompt issuance of the present application. Should any issue remain that prevents immediate issuance of the application, the Examiner is encouraged to contact the undersigned attorney to discuss the unresolved issue.

Respectfully Submitted,
Jed W. Caven
Caven & Aghevli LLC
9249 S. Broadway Blvd. #200-201
Highlands Ranch, CO 80129

Dated: 11/07/2005

Jed W. Caven
Caven & Aghevli LLC
Reg. No. 40,551
(720) 841-9544